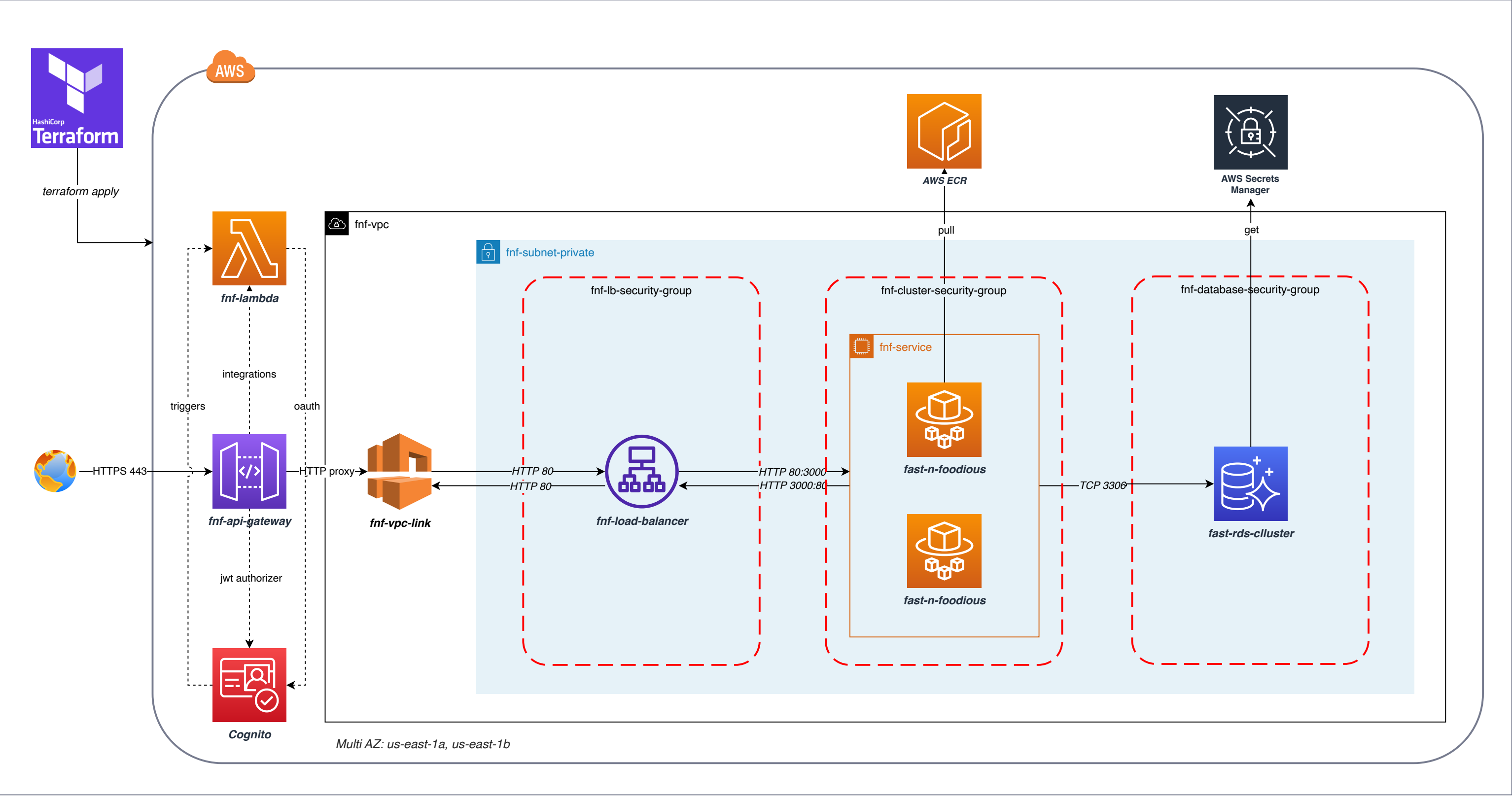
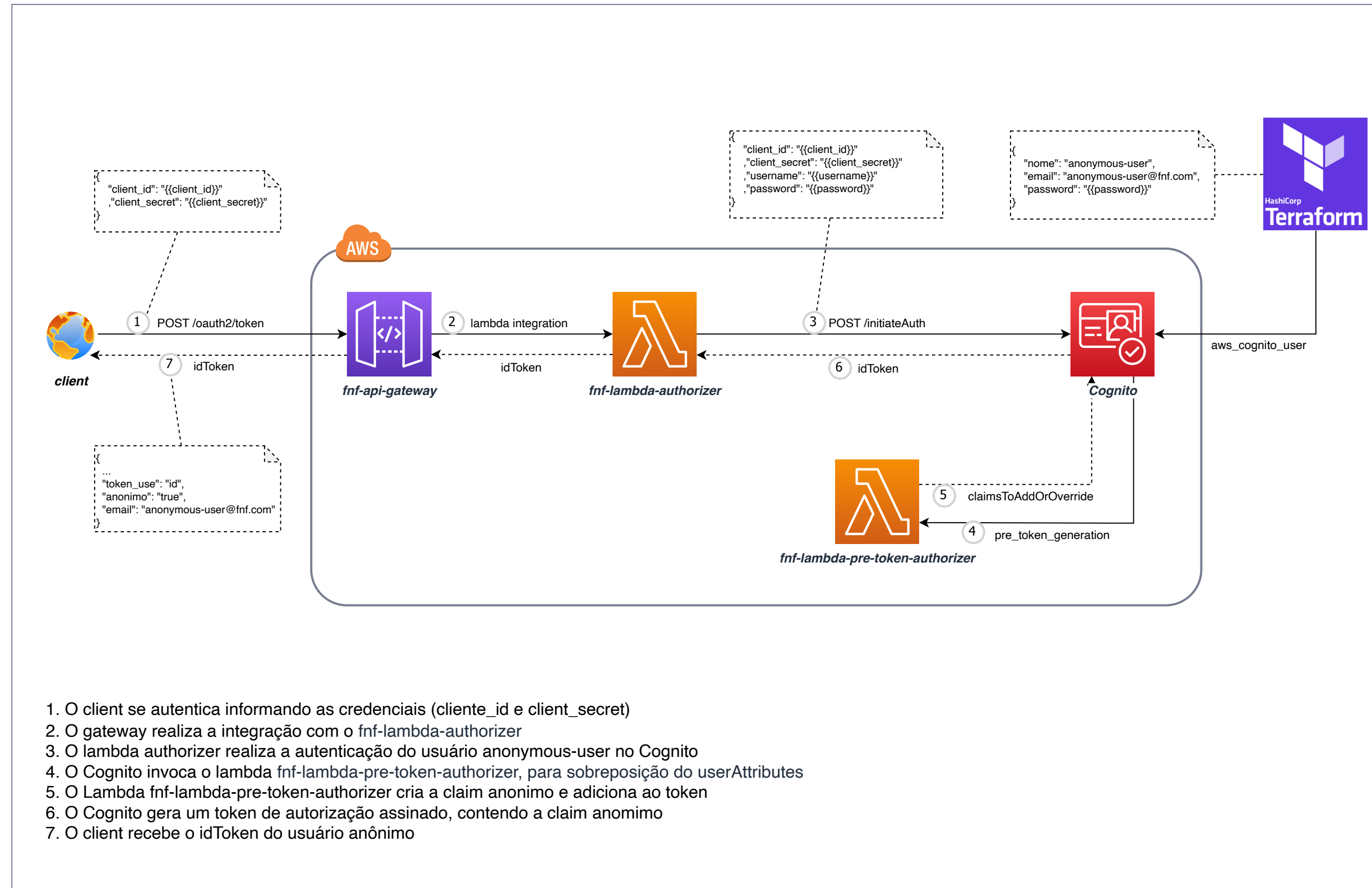


# Fast n' Foodious: Arquitetura Cloud AWS



# Fast n' Foodious: Autenticação de Usuário Anônimo



1. O client se autentica informando as credenciais (cliente\_id e client\_secret)
2. O gateway realiza a integração com o fnf-lambda-authorizer
3. O lambda authorizer realiza a autenticação do usuário anonymous-user no Cognito
4. O Cognito invoca o lambda fnf-lambda-pre-token-authorizer, para sobreposição do userAttributes
5. O Lambda fnf-lambda-pre-token-authorizer cria a claim anonimo e adiciona ao token
6. O Cognito gera um token de autorização assinado, contendo a claim anonimo
7. O client recebe o idToken do usuário anônimo

O diagrama ilustra a arquitetura de uma API Gateway com integração Lambda e Cognito. O fluxo é o seguinte:

- O cliente se autentica informando as credenciais (cliente\_id, client\_secret, username e password).
- O gateway realiza a integração com o fnf-lambda-authorizer.
- O lambda authorizer realiza a autenticação do cliente no Cognito.
- O lambda authorizer realiza a identificação do usuário (cliente fast-n-foodious), retornando dados de cpf, nome e email.
- O lambda authorizer realiza a autenticação do usuário identificado no Cognito.
- O Cognito invoca o lambda fnf-lambda-pre-token-authorizer, para sobreposição do userAttributes.
- O Lambda fnf-lambda-pre-token-authorizer cria a claim cpf e adiciona ao token.
- O Cognito gera um token de autorização assinado, contendo a claim cpf.
- O cliente recebe o idToken do usuário identificado.

Componentes e fluxos detalhados:

- client**: Interage com a API Gateway via POST /oauth2/token?cpf={{cpf}} (1) e recebe o idToken (6).
- fnf-api-gateway**: Recebe a requisição do cliente e realiza a integração com o fnf-lambda-authorizer (2).
- fnf-lambda-authorizer**: Realiza a autenticação do cliente no Cognito (3) e a identificação do usuário (4) via POST /v1/cliente/identifica?cpf={{cpf}}. Retorna o idToken (5) para o Cognito e o idToken (8) para o cliente.
- fast-rds-cluster**: Banco de dados que armazena as informações dos clientes. O fnf-lambda-authorizer consulta o banco com a query select \* from cliente where cpf={{cpf}} (4).
- fast-n-foodious**: Serviço que fornece dados de identificação do usuário.
- fnf-lambda-pre-token-authorizer**: Realiza a geração de tokens (6) e a sobreposição de userAttributes (7) no Cognito.
- Cognito**: Serviço de autenticação que gera o token (3) e o idToken (5).

Exemplos de payloads:

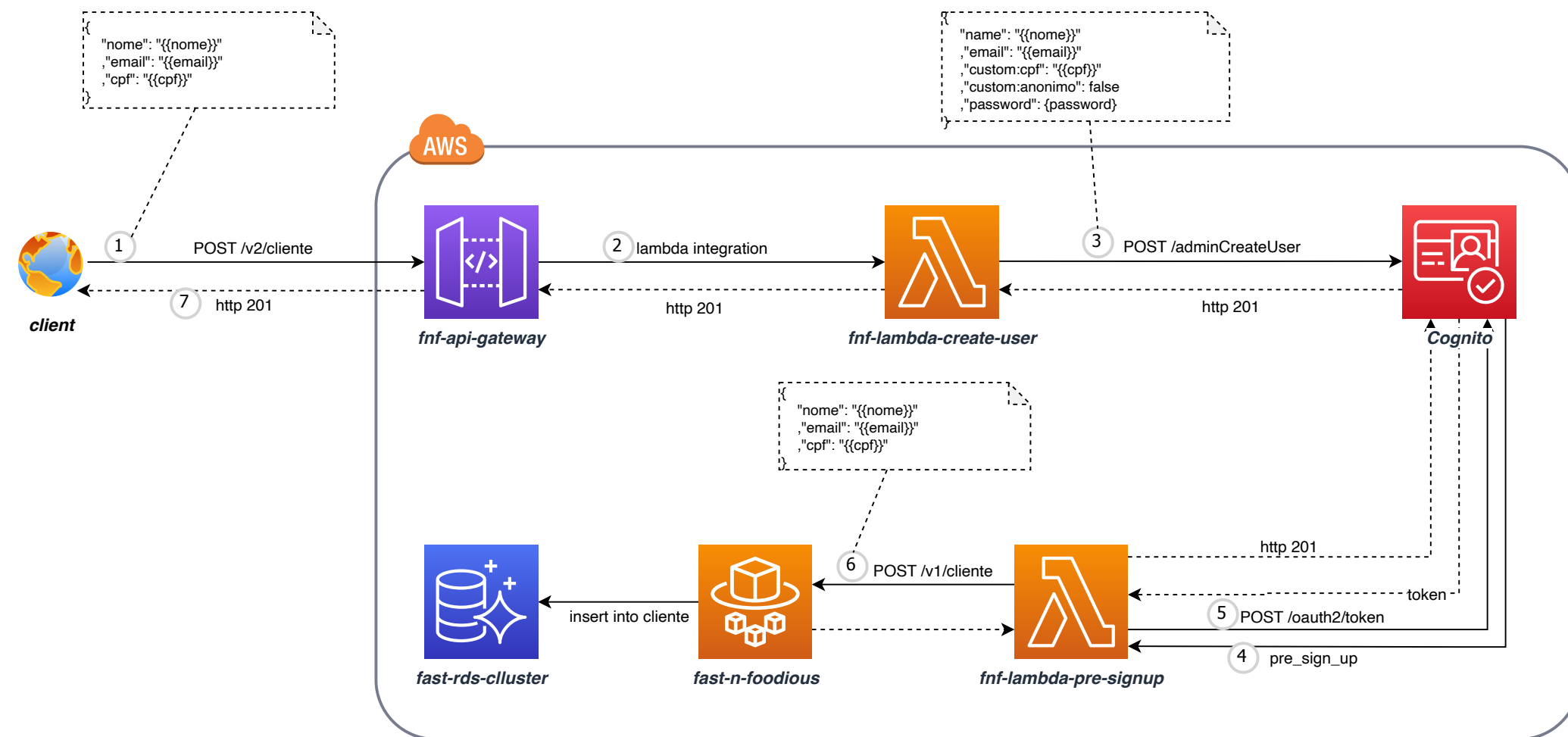
```

1. Client Credentials: {"client_id": "{{client_id}}", "client_secret": "{{client_secret}}"}
2. Client Identification: {"cpf": "{{cpf}}", "email": "{{email}}"}
3. Client Identification: {"client_id": "{{client_id}}", "client_secret": "{{client_secret}}", "username": "{{cpf}}", "password": "{{password}}"}
4. Client Identification: {"client_id": "{{client_id}}", "client_secret": "{{client_secret}}"}
5. Client Identification: {"client_id": "{{client_id}}", "client_secret": "{{client_secret}}"}
6. Client Identification: {"client_id": "{{client_id}}", "client_secret": "{{client_secret}}"}
7. Client Identification: {"client_id": "{{client_id}}", "client_secret": "{{client_secret}}"}
8. Client Identification: {"client_id": "{{client_id}}", "client_secret": "{{client_secret}}"}

```

1. O client se autentica informando as credenciais (cliente\_id, client\_secret, username e password)
2. O gateway realiza a integração com o fnf-lambda-authorizer
3. O lambda authorizer realiza a autenticação client no Cognito
4. O lambda authorizer realiza a identificação do usuário (cliente fast-n-foodious), retornando dados de cpf, nome e email
5. O lambda authorizer realiza a autenticação do usuário identificado no Cognito
6. O Cognito invoca o lambda fnf-lambda-pre-token-authorizer, para sobreposição do userAttributes
7. O Lambda fnf-lambda-pre-token-authorizer cria a claim cpf e adiciona ao token
8. O Cognito gera um token de autorização assinado, contendo a claim cpf
9. O client recebe o idToken do usuário identificado

# Fast n' Foodious: Cadastro de Usuários



1. O client realiza o cadastro de usuário, passando os dados de nome, email, cpf e senha
2. O gateway realiza a integração com o fnf-lambda-create-user
3. O lambda create user inicia o cadastro de novo usuário no Cognito
4. O Cognito invoca o lambda fnf-lambda-pre-singup, para inclusão de usuário na aplicação (cliente fast-n-foodious)
5. O lambda fnf-lambda-pre-singup realiza a autenticação client no Cognito
6. O lambda fnf-lambda-pre-singup realiza o cadastro do usuário na aplicação (cliente fast-n-foodious)
7. O client recebe http 201 como confirmação de usuário cadastrado